XML模式推断研究综述

郑黎晓[1,2],王 成[1]
(1. 华侨大学计算机科学与技术学院,福建厦门 361021; 2. 中国科学院软件研究所计算机科学国家重点实验室,北京 100190)

摘 要: 本文对XML(Extensible Markup Language)数据的模式推断问题研究现状与进展进行了阐述. 首先,从正规树文法的角度介绍了不同模式语言的理论模型. 进而从模式推断方法、目标模式语言、支持的表达能力、内容模型对应的正则表达式类型等多个方面对当前研究工作进行了细致的分类归纳和对比. 此外,还介绍了模式语言中支持的基本语义完整性约束推断的研究进展. 最后指出了当前研究中的不足,并对未来需要深入研究的方向进行了展望. 重在对XML模式推断的主流方法和前沿进展进行概括、比较和分析,以期对后续研究有所助益.

关键词: 可扩展标记语言; 模式推断; 正规树文法; 正则表达式
中图分类号: TP311 **文献标识码:** A **文章编号:** 0372-2112 (2016)02-0461-11
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2016.02.030

Schema Inference from XML Data: A Review

ZHENG Li-xiao[1,2], WANG Cheng[1]
(1. *College of Computer Science and Technology, Huaqiao University, Xiamen, Fujian* 361021, *China*;
2. *The State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing* 100190, *China*)

Abstract: This paper surveys the state of the art of schema inference from XML data. First, the formal models based on regular tree grammar for commonly used XML schema languages are presented. Then, the existing works on XML schema inference are summarized and compared from various aspects such as inference methods, target schema languages, supported expressiveness, regular expression types corresponding to the content models, and so on. In addition, inferences of some basic integrity constraints from XML data are also introduced. Finally, this paper points out the defects of current research and discusses some potential future research directions. This paper aims to offer a detail overview, comparison and analysis of the mainstream methods and recent progress in this field, expecting to be beneficial for subsequent research.

Key words: XML(Extensible Markup Language); schema inference; regular tree grammar; regular expression

1 引言

由万维网联盟(World Wide Web Consortium, W3C)在1997年提出的可扩展标记语言(Extensible Markup Language, XML)可以简单、灵活地描述各种带结构的数据,已经成为网络环境中数据交换与集成的事实标准,得到越来越广泛和深入的应用. 因此,如何有效处理和使用XML数据成为近年来一个持续的研究热点. 在XML数据处理中,模式是一个重要的方面. 模式定义了XML数据的结构、类型和语法规范,是保证XML数据格式正确和内容有效的手段. 模式在XML数据处理和应用程序中发挥着重要的作用,例如:用于XML文档的有效性验证和类型检查以保证应用程序的安全性和正确性[1];利用模式信息进行查询优化从而提高应用程序的实现效率[2];通过模式匹配和模式映射实现数据的自动集成和交换[3]等. 此外,一些软件开发工具如SUN JAXB等也都依赖模式实现XML数据绑定功能.

然而在现实中经常出现模式缺失或未有效定义的情况. 2006年的统计结果[4,5]显示互联网上可用的XML文档中近一半没有模式定义,而存在的模式中大约有2/3不满足W3C规范. 一项最新(2013)年的调查报告[6]指出这种情况变得越来越普遍:统计的数据中只有1/4的文档有模式定义,而这些模式中仅有1/3是有效的. 因此,如何从已有的XML文档中自动推断出符合规范的高质量模式成为一个亟待解决的问题. 实际上,早在2005年,W3C标准制定者之一Florescu[7]就强调

收稿日期:2014-07-02;修回日期:2014-08-26;责任编辑:马兰英
基金项目:华侨大学科研启动基金(No. 12BS215);国家自然科学基金(No. 51305142, No. 61502184);福建省自然科学基金(No. 2015J01259)

“在半结构化数据管理中,我们需要从已有的 XML 文档中自动抽取高质量的模式,并进行增量式维护”. 此外,模式推断在模式演化方面也有重要应用. 有些文档即使存在有效的模式,但定义可能过于泛化,即描述的约束信息相对于存在的 XML 文档而言过于宽泛. 实际上,这种情况在工业界普遍存在[8]. 此时,也需要利用模式推断算法获得更精确的描述,以更好地服务于应用.

XML 模式推断问题早在多年前就引起关注. 近年来,随着 XML 的广泛应用和模式缺失的普遍性,有越来越多的学者研究该问题. 本文在充分调研和深入研究的基础上对 XML 模式推断问题的研究进展进行了综述:(1)介绍常见 XML 模式语言及其理论模型,并基于此给出模式推断的形式化定义;(2)以语言的归纳学习经典理论模型为参照,将现有的推断方法归结为两大类进行阐述,从目标模式语言、支持的表达能力、内容模型对应的正则表达式类型等多个维度对不同的推断方法进行分析和对比. 此外,还介绍了模式语言中语义完整性约束推断的研究现状.

2 XML 模式语言及其理论模型

2.1 模式语言简介

现有的 XML 模式语言有很多种,其中较为常用的包括 W3C 推荐标准文档类型定义(Document Type Definition,DTD)[9] 和 XML 模式定义(XML Schema Definition,XSD)[10],以及国际标准化组织(International Organization for Standardization, ISO)推荐标准 RELAX NG[11]. 下面分别对这几种语言进行介绍,2.2 节将形式化地给出对应的理论模型.

DTD 是目前使用较为普遍、也是 W3C 最早推荐的 XML 模式语言. DTD 本质上是一种扩展的上下文无关文法. 在这种文法中,产生式的右部不是简单的字符串而是正则表达式,允许用户使用正则操作算子(|、*等)定义元素的出现顺序、次数等. 图 1(*a*)给出一个描述书本信息的 DTD 例子. 由于其简单易用,使得 DTD 得到了普遍的应用. 但在某些场合,DTD 有明显的局限性,例如基本类型较少,引用机制有限,缺少模块化,描述无序数据的繁琐等.

XSD 为弥补 DTD 的缺陷,W3C 于 2001 年推出了表达能力更强的模式定义语言 XSD. 与 DTD 相比,XSD 具有以下几大特点:(1)引入了类型的概念,允许使用相同的元素名定义不同的类型,例如,使用相同的“section”元素名来描述 Book 和 Paper 里面不同的章节类型;(2)允许使用“all”指示器指定子元素可以以任意顺序出现,即交互,和通过“minOccur”、“maxOccur”属性对元素的出现次数进行精确限定,即计数;(3)提供了类型派生机制,允许在已经定义的数据类型基础上,定义新的数据类型,可以实现类型复用和模式定义的紧凑性;(4)提供更丰富的内置数据类型,如 int、date、time 等. 此外,XML Schema 还支持命名空间的机制,实现了模式复用和避免命名冲突. 近年来,XSD 由于其更强的表达能力和丰富的定义机制正逐渐取代 DTD 而被广泛应用. 其缺点是定义较为复杂,可读性差. 例如,图 1(*c*)中的 XSD 和图 1(*a*)中 DTD 定义相同的信息,但是篇幅明显要长很多.

```
<!ELEMENT addrbook (person+)>
<!ELEMENT person ((name, tel?,email*)>
<!ELEMENT name (first, last)|(last, first)>
<!ELEMENT first (#PCDATA)>
<!ELEMENT last (#PCDATA)>
<!ELEMENT tel (#PCDATA)>
<!ELEMENT email (#PCDATA)>
```

(*a*) DTD

```
element addrbook{
 element person{
   element name{
     element first{text} &
     element last{text}
   }
   element tel{text} ?
   element email{text} *
   … …
 }+
}
```

(*b*) RELAX NG

```
<xs:schema xmlns:xs=
     "http://www.w3.org/2001/XMLSchema"/>
<xs:element name="addrbook">
 <xs:complexType>
   <xs:sequence>
     <xs:element ref="person" minOccurs="1"
                 maxOccurs="unbounded"/>
   </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="person">
… …
</xs:element>
<xs:element name="name">
 <xs:complexType>
  <xs:all>
    <xs:element name="first" type="xs:string"/>
    <xs:element name="last" type="xs:string"/>
  </xs:all>
 </xs:complexType>
</xs:element>
… …
```

(*c*) XSD

```
<schema xmlns:xs=
     "http://www.ascc.net/xml/schematron"/>
<pattern name="Conditions for addrbook">
 <rule context="addrbook">
   <assert test="person">
       there is no person
   </assert>
 </rule>
</pattern>
<pattern name="Conditions for person">
 <rule context="person">
   <assert test="name">
       there is no name for a person
   </assert>
   <report test="name[2]">
       there are multiple names for a person
   </assert>
   … …
 </rule>
</pattern>
 … …
</schema>
```

(*d*) Schematron

图1 XML模式语言示例

RELAX NG 最初由结构化信息标准促进组织(Organization for the Advancement of Structured Information Standards,OASIS)提出,现已成为 ISO 标准. 其初衷是为了结合 DTD 的简洁性和 XSD 的强表达能力,同时避免它们各自的缺陷. RELAX NG 具有两种定义方式:一种基于 XML 语法,提供了许多与 XSD 等同的功能,如子元素的

交互(即无序)出现、丰富的内置数据类型等;一种是紧凑的文本语法,使用元素嵌套的方式进行定义,实现与 DTD 类似的简洁性. 图 1(*b*)给出使用紧凑文法定义的一个例子. 由于 RELAX NG 对元素及其内容模型的定义没有规定严格的限制形式,因此其表达能力要强于 DTD 和 XSD(具体见 2.3 节).

其它　研究人员还提出了其他一些 XML 模式语言,例如 Schematron[12]、XDuce[13]、DSD[14] 等. 其中,Schematron 是一种基于规则的模式语言. 如图 1(*d*)所示,Schematron 使用断言的方式给出一系列规则描述 XML 文档中某个或某些元素或属性应该满足的约束. XML 数据的传统模型为有序无秩(ordered unranked)树. 最近,一些学者研究无序(unordered)树,提出描述无序 XML 树结构的模式语言 DMS[15]. DMS 采用与 DTD 类似的定义方式,特点是去除了正则表达式中描述出现顺序的连接符“,”,替换为无序连接符“‖”. 这些模式语言目前主要用于学术研究,实际应用还不太广泛.

2.2　理论模型

采用文法、自动机以及逻辑等方式[16,17] 都可以对 XML 的模式语言进行形式化描述. 由于 XML 文档既可以看成是字,也可以看作是树,因此描述其模式语言的文法或自动机也有两大类. 从字的角度上,有平衡文法[18] 以及可见下推语言(Visibly Pushdown Languages,简称 VPL)[19] 等. 其中平衡文法是由括号文法[20] 扩展而来,可以允许有多种形式的括号,且文法中产生式的右部允许使用正则表达式;VPL 是能被一个带有栈的下推自动机所接受的语言,在接受过程中根据输入的字母决定栈中对应的操作. XML 文档中的开始和结束标记对应于平衡文法中多种形式的括号,因此可以使用平衡文法描述 XML 的模式语言;标记也可以对应 VPL 中的配对符号,因此也可以利用 VPL 描述 XML 文档或进行 XML 相关的处理. 从树的角度上,最常见的描述 XML 模式语言的文法是正规树文法,其定义的语言是正规树语言,是能被树自动机所接受的语言[21]. 本节从正规树文法的角度进行介绍,详细内容参见文献[22].

2.2.1　正规树文法

一个正规树文法(Regular Tree Grammars,简称 RTG)是个四元组 $G=(V_N,V_T,S,P)$,其中 V_N、V_T 分别是非终极符和终极符的有限集合;$S\in V_N$ 是开始符号;P 是形如 $N\rightarrow lr$ 的产生式的有限集合,其中 $N\in V_N$,称为产生式的左部,lr 是产生式的右部,$l\in V_T$ 称为标号,r 是 V_N 上的正则表达式,称为产生式的内容模型(Content Model).

给定树 t 和正规树文法 G,称 t 是 G 的一个合法实例当且仅当存在一个映射 I 使得对于 t 中的任意结点 e,$I(e)$ 是 G 的一个非终极符并且满足下列条件:(1)若 e 是根结点,则 $I(e)$ 是开始符号;(2)对于结点 e 和其子结点 $e_1,\cdots,e_m$,存在 G 中的一条产生式 $X\rightarrow a\ r$ 使得 $I(e)$ 是非终极符 X,e 的标号是终极符 a,$I(e_1)\cdots I(e_m)$ 是正则表达式 r 的句子. 由 G 的所有合法树构成的集合称为 G 所定义的正规树语言.

对于正规树文法 G 中的两个非终极符 N_1 和 N_2,$N_1\neq N_2$,若存在两个产生式,其左部分别为 N_1 和 N_2,右部具有相同的终极符,则称非终极符 N_1 和 N_2 存在竞争. 如果 G 中不含有存在竞争的非终极符,则称 G 是 local 正规树文法;如果 G 中每条产生式的内容模型中的非终极符互不存在竞争,则称 G 是 single-type 正规树文法. 由以上定义可知,在 local 正规树文法中,非终极符与终极符是一一对应的. 换言之,由非终极符可以唯一确定相应的产生式和内容模型. local 的和 single-type 的正规树文法都是正规树文法的子类. local 的正规树文法也是 single-type 的,反之则不然. 因此,从表达能力上讲,三者之间的关系为:

$$\text{local RTG}\subset\text{single-type RTG}\subset\text{RTG}$$

2.2.2　正则表达式及其子类

正规树文法使用正则表达式定义产生式的内容模型,因此所描述的树均是无秩(unranked)的,即树中每个结点的子结点数目不是固定的. 这与 XML 数据的有序无秩树模型相吻合. 本节介绍与正则表达式及其子类相关的概念.

令 Σ 表示符号的集合. Σ 上的正则表达式(Regular Expression,简称 RE)可以递归的定义如下:空集$\varnothing$、空字符 ε、每个符号 $a\in\Sigma$ 是正则表达式;如果 E_1 和 E_2 是正则表达式,则将他们进行连接运算 E_1,E_2(为了方便表示,下文中省去了连接运算符)、选择运算 $E_1|E_2$、和星号运算 E_1* 的结果仍然是正则表达式. $L(E)$ 表示正则表达式 E 所定义的语言.

对于一个正则表达式 E,使用下标来依次标记其中出现的字符,使得每个标记后的字符在表达式中仅出现一次,这样的表达式称为 E 的标记表达式,记作 E'. 与标号操作相对应的是去标号. 假设 E 是一个带标号的正则表达式,去掉 E 中所有的标号后所得到的表达式记为 $E^{\#}$. 例如,令 $E=a*a$,则 $E'=a_1*a_2$,$(E')^{\#}=a*a$. 利用这些记号,我们可以定义正则表达式的一个特殊子类:one-unambiguous 表达式[23].

一个正则表达式 E 是 one-unambiguous 的,当且仅当对于任意两个句子 $uxv,uyw\in L(E')$,$|x|=|y|=1$,若 $x\neq y$ 则 $x^{\#}\neq y^{\#}$. One-unambiguous 正则表达式通常也被称为确定性(deterministic)正则表达式. 直观地讲,对于输入句子中的每一个符号,不需要向前看就能唯一地匹配确定性正则表达式中的一个位置. 例如,$a*a$ 不是确定性表达式,因为对于句子 a 而言,在不向前看的前提下不能确定符号‘a’与表达式中的第一个 a 还是第二个 a 匹配. 而 $aa*$ 是确定性表达式,对于 $L(aa*)$

中每一个句子,它的第一个字符都匹配 $aa*$ 中的第一个 a,其余的字符都匹配第二个 a.

在实际应用中使用更为广泛的一类表达式是带数字与交互的正则表达式,它们是对标准表达式的扩展,增加了带数字出现和带交互出现的操作符. 在 *XSD* 中使用的就是这类表达式,其定义如下:标准表达式是带计数和交互的表达式;令 E_1 和 E_2 是带计数与交互的表达式,则 $E_1^{[m,n]}$、$E_1^{[m,\infty]}$ 和 $E_1\&E_2$ 是带数字与交互的表达式(其中,∞ 表示无穷).

计数操作精确限定了字符允许出现的次数,交互操作则允许字符的交互出现. 例如 $L(a^{[1,3]})=\{a,aa,aaa\}$,$L(a\&b)=\{ab,ba\}$. 类似地,可以定义带计数与交互的确定性表达式.

2.3 与模式语言的对应关系

本节讨论几种常见 XML 模式语言对应的具体理论模型.

(1)表达能力方面

在 DTD 中,元素的类型声明对应于正规树文法中的一条产生式,元素的内容模型对应产生式的内容模型,元素名称既充当终极符又充当非终极符的角色. 这刚好与 local 正规树文法的定义形式吻合. 因此,从表达能力上讲,DTD 属于 local 正规树文法. XSD 中的主要特性如复杂类型定义、抽象类型定义、类型派生等均可以转换为正规树文法的形式描述. 特别地,W3C 规范要求对于每一个元素,根据其上下文即可确定其类型. 换言之,处于同一个内容模型中的相同元素名不能够存在竞争. 因此,XSD 的表达能力属于 single-type 正规树文法. RELAX NG 对元素的定义形式没有限制,表达能力最强. 本质上,RELAX NG 能够表示任何正规树语言.

(2)内容模型方面

W3C 规范要求 DTD 和 XSD 的元素内容模型需要满足惟一粒子属性(Unique Particle Attribution,UPA)约束,即表达式应该是确定性的. 其中,DTD 使用标准的确定性表达式,XSD 中支持计数和交互定义机制,使用的是带计数和交互的确定性表达式. RELAX NG 中没有确定性的要求,并且只支持交互定义机制,因此内容模型是带交互的表达式.

表 1 给出上述模式语言对应的具体理论模型. 其中 RE 和 dRE 分别表示标准和确定性正则表达式,#和 & 分别表示计数和交互.

表 1 XML 模式语言的理论模型

模式语言	理论模型	
	表达能力	内容模型
DTD	local RTG	dRE
XSD	single-type RTG	dRE(#,&)
RELAX NG	RTG	RE(&)

3 XML 模式推断问题定义

XML 模式推断属于语言的归纳学习问题,研究如何从语言的有限信息出发,通过归纳推断得到语言的定义. Gold 于 1969 年给出语言学习的经典理论模型,即语言的极限识认模型[24]. 按照该模型,XML 模式推断问题形式化地定义如下:称一组 XML 文档为一个样本,模式推断是一个从样本集到目标模式语言的映射 $\mathscr{M}$,使得:(1)对于每一个样本 D,$D\subseteq L(\mathscr{M}(D))$;(2)对于每一个目标模式语言中的模式定义 $\mathscr{S}$,存在一个样本 Dc,使得对于任意满足 $Dc\subseteq D\subseteq L(\mathscr{S})$ 的样本 D 都有 $\mathscr{M}(D)=\mathscr{S}$.

直观上来讲,条件(1)强调推断是正确的(sound),条件(2)则强调推断是完全的(complete). 显然,推断问题与目标模式语言有关. 目前的研究主要集中于 DTD 和 XSD,而 RELAX NG 的研究较少. DTD 属于 local 正规树文法,内容模型与元素是一一对应的,因此本质上归约为确定性正则表达式的学习;XSD 引入了类型的概念,因此推断包括两个方面:垂直方向——支持 single-type 正规树文法表达能力的类型识别;水平方向——支持计数与交互定义机制的内容模型推断. 其中后者又归约为带交互与计数的确定性表达式学习.

实际上,Gold 证明出如果只有正例,任何包含所有有限语言和任一无限语言的语言类均是无法识认的. 这也意味着正规树文法及其 local、single-type 子类都是无法学习的,甚至连内容模型对应的正则表达式及其确定性子类也无法学习[25]. 因此,现有的研究主要从两个方面入手:(1)只关注推断的正确性,对目标语言类和推断的完全性没有进行讨论,我们称之为启发式的推断方法;(2)是通过对目标语言加以限制,从而寻找到正确性与完全性兼顾的学习算法,我们称之为基于语言极限识认模型的推断方法.

表 2 给出了当前主流 XML 模式推断研究的具体分类结果,从推断方法、目标模式语言、支持的表达能力、内容模型对应的正则表达式类型等多个维度进行分析和对比.

4 启发式的推断方法

不管目标模式语言是 DTD 还是 XSD,模式推断的本质是从输入样本集中推导正规树文法. 因此,一般而言,启发式的推断方法主要分为如下几个步骤:(1)推导初始文法;(2)推导正则表达式;(3)转换为目标模式语言. 其中,第(1)步所采取的策略决定了推断方法所支持的表达能力:是 local 正规树文法还是 single-type 正规树文法等? 第(2)步则影响推断结果中内容模型对应的正则表达式类型:是标准表达式还是带计数或交互的扩展表达式? 第(3)步则涉及到具体的目标模式语言.

4.1 推导初始文法

首先,对于 XML 文档中的每一个元素 e 和其子元素 $e_1,\cdots,e_k$,生成一条产生式 $e\rightarrow lab(e)\ e_1,\cdots,e_k$,其中,$lab(e)$是元素 e 的标号. 由于对每一个元素都生成了一条产生式,接下来需要对产生式进行聚类以识别出相同的元素类型. 大多数文献如[26,27]等采用的方式是将产生式左部符号相同的归为一类. 显然,这种方式将所有标号相同的元素识别为同一类型,得到的是 local 正规树文法(也即 DTD). 文献[28,29]提出除了元素名之外,还利用元素到根的路径信息进行产生式聚类,即将所有祖先路径相同且产生式左部符号也相同的产生式归为一类,从而保证获得 single-type 正规树文法(即 XSD)的表达能力. 文献[28]指出,为了获得正规树文法的表达能力,不能简单采取祖先路径的方式聚类. 该文献进一步提出,对于相同标号的元素,可通过计算其子树之间的结构相似度来归类. 子树之间的相似度根据树上的编辑距离[30]来衡量,产生式聚类利用 MNC(Mutual Neighborhood Clustering)等聚类算法[31]实现.

4.2 推导正则表达式

推导正则表达式是整个推断过程中最重要的一步. 常用的是一种称作“合并前缀树自动机状态”的推导方法:首先对每一个产生式聚类,根据其右部的符号串构造前缀树自动机(Prefix Tree Automata,简称 PTA);然后按照一定的泛化规则对自动机进行状态合并. 假设符号串集合为 STR,则从 STR 中构造前缀树自动机的方式为:对于 STR 中的第一个串,构造一个接受该串的简单自动机;对于 STR 中剩余的每一个串,尽可能多的顺着自动机的迁移规则匹配串中的字符,如果遇到一个字符不能匹配,则在自动机中添加新的路径来接受剩余的串. 图 2 给出一个从含有两条产生式的聚类中构造前缀树自动机的例子.

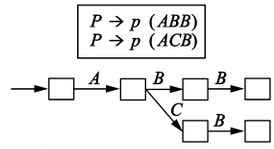


图2　前缀树自动机示例

文献[32]采用一组较为简单的规则对前缀树自动机的状态进行合并,这些规则都是基于经验的推测,例如:$aa\Rightarrow a+$,表示若一个字符出现了两次,则很可能该字符允许出现任意次. 类似的规则还有:$a?\ b?\ c?\Rightarrow a|b|c$,$a\ b\ c\ d\ a\ d\ b\ c\Rightarrow(a|b|c|d)+$等. 对前缀树自动机不断作用状态合并规则,一直到没有可用的规则时停止. 显然,合并的过程对自动机所定义的正则语言进行了泛化,并且规则的不同使用顺序等因素也影响了最终推导结果的不确定性. 为了选择较优的结果,文献[26]提出使用最小描述长度(Minimum Description Length,MDL)原则[33]从所有可能的候选表达式中进行选择. MDL 原则从表达式的紧凑度(conciseness)和精确度(preciseness)两方面进行评估,前者指存储表达式本身所需的位数,后者指使用该表达式解释输入串所需的位数. 显然,这种方法的优点是能够推导出较优的结果,缺点是受搜索空间影响,效率可能降低. 文献[34]根据前缀树自动机中状态的等价性来进行合并. 称状态 s_1、s_2等价当且仅当所有从 s_1、s_2出发到达某一终止状态的路径都相同. 由于检查状态等价的复杂度较高,实际应用中可以将路径长度限定为某个 $k(k\geqslant 1)$值. 同时,该文献还引入蚁群算法[35]的思想来搜索最优解.

表 2　XML 模式推断方法分类对比

方法类别	文献	目标模式语言			表达能力			内容模型			
		DTD	XSD	其他	local	single-type	RTG	RE	dRE	RE(#)	RE(&)
启发式	26	√			√			√			
	27	√			√			√			
	28		√			√	√				√
	29		√			√				√	
	32	√			√			√			
	34	√			√			√			
	36		√		√					√	
	37		√		√					√	
	38	√	√	√	√				√		
	39		√		√			√			
	40			√							
	41			√							
基于语言极限认识模型	45	√			√				√		
	46	√			√				√		
	48	√			√				√		
	50	√			√				√		
	51		√			√			√		

XSD 允许元素以任意顺序出现和对元素的出现次数进行精确限定,因此其内容模型是带交互和计数的确定性表达式. 文献[28]和文献[29,36,37]在推导表达式时分别考虑了对“交互”和“计数”操作的支持. 同样地,推导规则仍旧是启发式的,例如通过统计元素实际出现的最小和最大次数来确定“计数”操作中的 *minOccur* 和 *maxOccur*. 为进一步提高表达式的可读性和紧凑性,许多文献还包括了对推导得到的表达式进行结构化简的步骤. 化简的原则是保持语言的等价性,有时也允许适当泛化. 常见的一些化简规则包括:$a?? \Rightarrow a?$,$a?\ a+ \Rightarrow a*$,$(a,b)|(a,c) \Rightarrow a\ (b|c)$等. 一些文献将化简与表达式推导融合在一起,边推导边化简. 部分文献如[29]没有考虑这一步骤.

除了文献[38]外,上述提及的表达式推导过程均没有考虑确定性问题,因此推导结果仍旧是标准表达式. 严格地讲,并不符合 W3C 规范中对 DTD 和 XSD 的元素内容模型需要满足惟一粒子属性约束的要求.

4.3 转换为目标模式

最后一步是将以树文法形式描述的推断结果转换为目标模式语言. 由正规树文法翻译为 DTD 比较直观,而转换为 XSD 则较为复杂. 文献[39]讨论了由推断结果转换成 XSD 语法时遵循的实用规则,如元素的内容模型应该直接在该元素定义中给出,还是另外定义一个新的复杂类型;如何识别类型间的派生(extension、restriction)关系等? 文献[38]则缺省了该步,由用户根据需要自由转换为所需的目标模式.

与 DTD 中只支持 PCDATA 数据类型不同,XSD 中都支持丰富的基本类型和用户自定义类型. 因此,转换为 XSD 时还需要进行数据类型的识别. 目前,只有文献[37,38]提到这一问题,并且只讨论了简单的数值类型如 decimal、int 以及字符串 string 等类型的识别.

4.4 其他模式语言的推断

Schematron 使用断言的方式给出一系列规则,不仅能够定义 XML 文档的结构约束,还能够定义较为复杂的语义完整性约束. 文献[40]研究如何由正规树文法生成 Schematron 规则,因此解决的仍旧是结构约束方面的推断,没有涉及较复杂的语义约束. 文献[41]对学术界最近提出的描述无序 XML 树的模式语言 DMS 的推断问题进行初步探讨. 这两个文献中的推断方法没有参照语言学习的理论模型,仍旧属于启发式的推断.

4.5 优缺点分析

在启发式的推断方法中,无法对推断结果所属的语言类或文法类进行定义,也无法从理论上保证算法的学习能力等性质. 然而,由于其直观、灵活、易实现等特点,在当前仍有较广泛的研究和应用. 文献[42]对该类推断方法进行了深入探讨,文献[43]实现一个启发式推断方法研究的辅助工具集,提供文档解析、自动机可视化等基本功能.

5 基于语言极限识认模型的推断方法

内容模型推断本质上归结为确定性表达式的学习. Gold 定理已指出:不能在有限时间内仅通过正样例来识认类别不受限制的正则表达式. Bex 等人[25]证明出这一结论同样适用于确定性表达式. 因此,在基于语言极限识认模型的推断方法中,首先需要寻找可学习的受限表达式子类,其次,以受限表达式为目标语言,设计具有正确性和完全性的学习算法.

5.1 受限表达式子类

确定性表达式可学习子类中最具代表性的两个分别是单次出现表达式和链表达式. 一个正则表达式中如果每个字符最多只出现一次,那么称该表达式为单次出现表达式(Single Occurrence Regular Expression,简称 SORE). 例如,$(((a^+(b|c)?)^+)d)^+$是一个 SORE,而 $a?(ba|c)$不是一个 SORE. 如果一个 SORE 满足如下形式:$f_1\cdot\cdots\cdot f_n(n\geqslant 0)$,其中每个 f_i是一个形如$(a_1|\cdots|a_k)$、$(a_1|\cdots|a_k)?$、$(a_1|\cdots|a_k)^+$或$(a_1|\cdots|a_k)^+?$的链式因子,其中 $k\geqslant 1$,每个 $a_j(1\leqslant j\leqslant k)$都是一个终极符,那么称该正则表达式为链表达式(Chain Regular Expression,简称 CHARE). 例如,$(a|b)^+?\ c?\ (d|e|f)^+$是一个 CHARE,$(ab^+|c|d)?\ (e?\ |f|g^+)^+$则不是.

根据定义可知 SORE 和 CHARE 属于确定性正则表达式. 文献[44]中的统计结果显示,XML 模式中出现的正则表达式绝大部分为 SORE,而 SORE 中 90% 为 CHARE. 文献[45,46]证明出对于这两类受限表达式,存在满足 Gold 极限识认模型的推断算法,即存在一个既是正确的又是完全的推断算法. 算法分为两步:首先根据输入串构造单次出现自动机(Single Occurrence Automata,简称 SOA),其次从 SOA 中推导受限表达式.

5.2 构造 SOA

令 Σ 表示一个字母表,*src* 和 *sink* 是两个特殊符号,*src* 表示开始状态,*sink* 表示接受状态. Σ 上的单次出现自动机 SOA 是一个满足以下两个条件的有向图 $A=(V,E)$:(1)$\{src,sink\}\subseteq V, V\subseteq\Sigma\cup\{src,sink\}$,$V$ 中的每个结点都有结点标记,对应于 Σ 中的一个终极符;(2)*src* 结点只有出边,*sink* 结点只有入边,每一个结点 $v\in V$ 都位于 *src* 到 *sink* 的路径上. 一个句子 $w=a_1\cdots a_n(n\geqslant 0)$被 SOA 接受,当且仅当在 SOA 中存在这样一条路径:$src\rightarrow v_1\rightarrow\cdots\rightarrow v_n\rightarrow sink$,其中结点 v_i的标记为 $a_i(1\leqslant i\leqslant n)$. $L(A)$表示所有被 SOA $A=(V,E)$接受的句子集合,即被 A 接受的语言.

SOA 的定义和通用的自动机定义有些区别,SOA 中的边没有迁移标记,每个结点(状态)都对应于一个

终极符,称为结点标记. 不难看出 SOA 是确定性有穷自动机(Deterministic Finite Automaton,简称 DFA)的子类,且每个 SOA 都存在一个等价的 DFA 与之对应. 如果在 DFA 中存在一条迁移标记为 $a \in \Sigma$ 的边指向状态 q,那么相应地在 SOA 中存在一个标记为 a 的结点有一条入边. DFA 中的开始状态和结束状态分别对应于 SOA 中的 *src* 结点和所有到 *sink* 存在迁移边的结点.

从句子集 S 构造 SOA 可采用经典的 2T-INF 算法[47]:如果非终极符 a、b 在句子集 S 中是相邻的,则称 (a,b) 为 S 的一个二元组;对于 S 的每一个二元组 (a,b),添加一条从顶点 a 到顶点 b 的边,便可构造出输入串对应的 SOA. 例如,句子集合 $S=\{aab,cdd\}$,由 2T-INF 生成的 SOA 如图 3 所示,其中 *src* 和 *sink* 分别表示起始点和结束点.

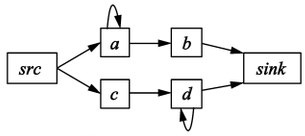


图3　构造SOA

显然,对于给定句子集合 S 和由 2T-INF 算法生成的 SOA A ,有 $S \subseteq L(A)$ 成立. 文献[46]指出,对于目标表达式 E 和句子集 S,如果 E 所定义的语言 $L(E)$ 中的所有二元组 (a,b) 都在句子集 S 中出现,则 $L(E)=L(2T\text{-}INF(S))$.

5.3　推导受限表达式

文献[46,48]分别给出由 SOA 推导受限表达式的两种不同方法. 文献[46]中的推导是基于一组重写规则,例如:若 SOA 中两个顶点 a 和 b 具有相同的前驱顶点和相同的后继顶点,则将其重写为一个顶点 $a|b$;若顶点 a 到自身有一条边,则将 a 及其反身边重写为一个顶点 a^+. 在 SOA 上不断地作用重写规则,一直到没有适用的规则时停止,最后得到的顶点即是 SORE. 由 SOA 推导 CHARE 则是基于有向无环图的拓扑序列. 首先计算出 SOA 中的强连通分量,将其替换为一个顶点:若分量中含有三个顶点 a、b、c,则替换为顶点为 $(a|b|c)^+$. 替换后的 SOA 是一个有向无环图,求出其任一拓扑序列,使用连接算子连接起来即是推导得到的 CHARE. 文献[46]证明出对于任一 SORE(或 CHARE) E,存在一个输入集 S,使得 $L(E)=L(\tau(2\text{T-INF}(S)))$,其中 τ 表示上述推导算法. 因此,这也意味着当输入信息足够多时,推断得到的 SORE 表达式不再变化,满足语言极限识认模型中的"完全性"要求. 例如,图 3 所示的 SOA 推导得到的 SORE 和 CHARE 分别为:$(a^+b)|(cd^+)$、$a^+?\ c?\ b?\ d^+?$.

语言的归纳学习存在泛化问题,比较理想的状态是实现最小包含泛化或称描述性泛化[49]. 对于输入串 S 和推导得到的 SORE(或 CHARE)表达式 E,如果不存在一个 SORE(或 CHARE)表达式 E' 满足 $S \subseteq L(E') \subseteq L(E)$,则称 E 是 S 的 SORE(或 CHARE)-最小包含泛化. 文献[48]指出文献[46]中的推导算法虽然满足语言极限识认模型,但是没有达到描述性泛化要求,因此不是最优的. 该文献提出对 SOA 中的顶点划分级数,结合顶点所处的级数和其它相关信息施加相应的表达式算子进行推导,并证明了算法的最小包含泛化特性. 在某些情况下,两种推导算法得到的表达式相同. 也就是说文献[46]的算法在某些情况下也可以实现最小包含泛化,但不是对所有情况都成立.

5.4　k-ORE 和 k-local XSD

文献[50]将表达式的限制放松为每个字符可以出现 k 次,称为 k-ORE. 对于 k-ORE,虽然存在满足 Gold 极限识认模型的学习算法,但是算法的运行时间为指数级. 为此,该文献给出另外一种基于隐马尔科夫模型的学习算法,首先利用概率模型构造 kOA,再转换为 k-ORE. 对于同一个输入集和不同的 k 值,可以先学习出若干个 k-ORE,然后利用最小描述长度原则选择最优者.

SORE、CHARE、以及 k-ORE 的学习算法均可应用于 XSD 的内容模型推断. 除此之外,需要解决的另一个问题是 XSD 的类型识别. Bex 等[51]发现在大部分 XSD 中,元素类型由其最近的祖先名称所决定,据此提出 k-local XSD 的概念并给出相应推断算法. 显然,该方法的一个缺点是推断结果与 k 的选择有关. 实际上,当 $k=1$ 时即退化为 DTD. 另一个缺点是会产生大量的类型,导致最终得到的模式定义不够简洁.

5.5　优缺点分析

基于语言极限识认模型的推断方法的优点是推断结果所属的语言类具有清晰明确的界定,推断算法从理论上可以保证具备良好特性:既满足正确性又满足完全性,甚至最小泛化特性. 缺点是推断结果受到限制,比如内容模型必须是 SORE 或 CHARE 等,缺乏灵活性.

6　基本的语义完整性约束推断

除了结构约束外,一般模式语言还提供了基本的语义完整性约束定义机制. 例如,W3C 推荐在 XSD 中使用"key"和"keyref"元素定义主键和外键,许多行业标准采用这些约束描述数据的联系与依存规则. DTD 中提供的"ID"和"IDREF"属性类型起到类似作用,但是定义能力相对较弱. 键约束是最基本也最常见的完整性约束类型,用于保证实体完整性和引用正确性,在查询优化、数据质量管理、数据交换中保持语义信息等

方面发挥着重要作用. 因此,完整的模式推断不仅需要考虑结构约束,还需要考虑语义方面的约束. 相比于结构约束,XML 的完整性约束推断算法研究较少.

(1)DTD 中的约束推断

DTD 中提供了 ID/IDREF(S)属性类型用于在整个文档中标识元素和定义引用,类似于主键和外键. 寻找 XML 文档中 ID/IDREF(S)属性的基本步骤是首先计算出元素和属性之间的映射关系,然后根据是否满足单射来确定元素的 ID 属性,最后依据 ID 属性值来寻找 IDREF(S)属性. 为了避免产生无意义的属性,文献[52]引入了"支持度"和"覆盖度"的概念来衡量和选择最优者. 文献[53]使用类似方法,不同之处在于将选择最优者归约为整数线性规划解决.

(2)XSD 中的约束推断

XSD 中的"key"和"keyref"元素使用路径表达式定义主键和外键约束,允许约束限定在文档的某个范围内,比 DTD 中的 ID/IDREF(S) 的表达能力强. XSD 主键定义形式化描述为 $key=(c,(Q,\{P_1,P_2,\cdots,P_k\}))$,其中 c 称为上下文,由元素名和该元素的类型决定;Q 和 $\{P_1,P_2,\cdots,P_k\}$ 分别称为目标路径和键路径集,使用一类受限 XPath 表达式定义. XSD 规范要求对于每一个键路径表达式 P_i 和由 Q 计算得到的作用域中的每一个目标结点 u 而言,$P_i(u)$ 的计算结果必须是 singleton 的,即只能包含一个带有数据值的结点. XSD 规范要求键定义应该与模式定义相一致,也即对于任意符合模式的 XML 文档,键都应满足上述规范. 外键约束定义形式与主键类似,键路径表达式也需满足相同的规范. 此外,XSD 还限定被引用的主键应与外键在同一上下文范围内.

目前为止只有文献[54]探讨了 XSD 主键约束的推断算法. 该文献首先采用逐层搜索(levelwise search)[55]方法根据 XML 文档中的信息找出所有合法的且满足一定支持度的候选主键,然后进行一致性检查去除与模式不一致的,最后利用关系数据上的函数依赖发现算法[56~58]验证最终的主键. 受搜索空间与一致性检查等影响,该方法在效率上存在缺陷. 实际上,该文献给出的实验数据显示,当输入规模较大时,几乎全部的时间消耗在一致性检查上,而最终得到的一致性主键只占整个候选主键的很小一部分. 该文献也指出如何解决这一瓶颈是个重要的遗留问题.

(3)其他约束推断

较早关于 XML 键的定义由 Buneman 等[59]给出,定义方式与 XSD 相同,但是没有限制与模式定义的一致性. 文献[60~63]针对 Buneman 等的键定义研究自动发现算法. 其中,文献[60]利用关系数据上的相关算法实现,文献[61,62]关注发现近似键,文献[63]则研究从 XQuery 查询日志中寻找. 此外,主键和外键分别是函数依赖和包含依赖的特例. 一些学者研究 XML 函数依赖及其发现算法[64~67],但是同样没有考虑模式问题.

7 结束语

本文针对 XML 模式推断问题的研究现状进行了较为全面的介绍和讨论:首先,从树文法的角度介绍了不同模式语言的理论模型;其次,将现有的模式推断方法分为启发式的方法和基于语言极限识认模型的方法两大类,并且从目标模式语言、支持的表达能力、内容模型对应的正则表达式类型等多个维度进行分析和讨论;最后,介绍了模式语言中支持的基本语义完整性约束的推断研究.

XML 模式推断的重要性与现实必要性引起越来越多的学者的关注. 然而,XML 半结构化的模型、复杂的模式定义等,给问题的解决增加了难度,尚有许多值得深入探索的问题. 在本文的最后,我们提出一些值得进一步研究和探讨的方面.

(1)正反例相结合的推断. Gold 定理指出在只有正例的情况下,即便是表达能力较弱的正则语言也无法学习. 如果在提供正例的同时还提供反例,则有望克服基于语言极限识认模型的推断方法在仅有正例时的局限性. 但是在实际应用中,由于反例信息不容易自动辨识或获取,因此可能需要进行人为的干预.

(2)输入数据预处理. 网络环境的复杂性极易导致噪声数据的存在. 例如,一组描述某公司订单信息的 XML 文档中可能不小心掺杂个别关于其他信息的文档. 噪声数据的模式一般会显著区别于其他的情况,如果不进行处理,会导致推断得到的模式过于一般化. 为获得更加精确和有效的模式,有必要对输入数据进行噪声消除的预处理.

(3)对 XSD 中高级类型定义机制的支持. 已有的 XSD 推断研究大多利用了 DTD 的内容模型生成方法,因此得到的 XSD 在定义能力上仍旧局限于 DTD. 如何有效支持 XSD 中的继承、多态、用户自定义类型等定义机制需要进一步的研究.

(4)增量式维护. 随着网络的广泛普及和应用的深入,网络数据往往呈现快速增长的趋势. 增量计算是应付数据动态变化和降低计算成本的有效手段. 因此,有必要研究对推断的模式进行增量维护的方法,以减少重复计算,提升处理效率,更好地满足实际应用的需求.

(5)完整性约束推断. 除了结构约束外,一般模式语言还提供了基本的语义完整性约束定义机制. 因此,完整的模式推断不仅需要考虑结构约束,还需要考虑语义方面的约束. (1)现有的研究主要关注前者而忽略

了后者;(2)当前大多数完整性约束发现算法与模式完全独立,而 W3C 规范要求某些约束,如 XSD 键等,应该与模式定义相一致. 因此,如何结合模式信息推断符合规范的复杂语义约束也是一个重要研究方向.

参考文献

[1] Tova M, Suciu D, Vianu V. Typechecking for XML transformers[J]. Journal of Computer and System Sciences, 2003,66(1):66 – 97.

[2] Bjørklund H, Martens W, Schwentick T. Optimizing conjunctive queries over trees using schema information[A]. Proceedings of the International Symposium on Mathematical Foundations of Computer Science[C]. Berlin Heidelberg:Springer,2008. 132 – 143.

[3] Halevy A, Rajaraman A, Ordille J. Data integration: the teenage years[A]. Proceedings of the VLDB[C]. New York:ACM Press,2006. 9 – 16.

[4] Barbosa D, Mignet L, Veltri P. Studying the XML web: Gathering statistics from an XML sample[J]. World Wide Web,2006,9(2):187 – 212.

[5] Martens W, Neven F, Schwentick T and Bex G J. Expressiveness and complexity of XML Schema[J]. ACM Transactions on Database Systems,2006,31(3):770 – 813.

[6] Grijzenhout S, Marx M. The quality of the XML web[J]. Web Semantics: Science, Services and Agents on the World Wide Web, March 2013,19:59 – 68.

[7] Florescu D. Managing semi-structured data[J]. ACM Queue,2005,3(8):18 – 24.

[8] Hinkelman S. Business integration- – Information conformance statements[R]. Technique Report, IBM Developer Works,2005.

[9] Bray T, Paoli J, et al. Extensible markup language(XML) 1.0(Fifth Edition)[EB/OL]. http://www.w3.org/TR/2008/REC-xml-20081126/,2008.

[10] Gao S, Sperberg-McQueen C M, Thompson H S. XML schema definition language(XSD)1.1 part 1: structures[EB/OL]. http://www.w3.org/TR/2012/REC-xmlschema11-1-20120405/,2012.

[11] Clark J, Murata M. RELAX NG specication[EB/OL]. http://www.oasis-open.org/-committees/relax-ng/,2001.

[12] Jelliffe R. The schematron——an XML structure validation language using patterns in trees[EB/OL]. http://xml.ascc.net/resource/schematron/,2001.

[13] Hosoya H, Pierce B C. XDuce: A statically typed XML processing language[J]. ACM Transactions on Internet Technology(TOIT),2003,3(2):117 – 148.

[14] Klarlund N, MØller A, Schwartzbach M I. DSD: A schema language for XML[A]. Proceedings of the Third Workshop on Formal Methods in Software Practice[C]. New York:ACM Press,2000. 101 – 111.

[15] Boneva I, Ciucanu R, Staworko S. Simple schemas for unordered XML[A]. Proceedings of the WebDB[C]. New Tork:ACM Press,2013. 13 – 18.

[16] Berstel J, Boasson L. XML grammars[A]. Proceedings of the International Symposium on Mathematical Foundations of Computer Science[C]. Berlin Heidelberg: Springer, 2000. 182 – 191.

[17] Neven F. Automata, logic, and XML[A]. Proceedings of the Conference of the European Association For Computer Science Logic[C]. Berlin Heidelberg: Springer, 2002. 2 – 26.

[18] Berstel Jean, Boasson L. Balanced grammars and their languages[J]. Formal and natural computing, 2002, 2300(3):3 – 25.

[19] Alur R, Madhusudan P. Visibly pushdown languages[A]. Proceedings of the STOC[C]. New York: ACM Press, 2004. 202 – 211.

[20] McNaughton R. Parenthesis grammars[J]. Journal of the ACM,1967,14(3):490 – 500.

[21] Comon H, Dauchet M, et al. Tree automata techniques and applications[DB/OL]. Available on: http://www.grappa.univ – lille3.fr/tata,2012.

[22] Murata M, et al. Taxonomy of XML schema languages using formal language theory[J]. ACM Transactions on Internet Technology,2005,5(4):660 – 704.

[23] Brüggemann-Klein A, Wood D. One-unambiguous regular languages[J]. Information and computation, 1998, 140(2):229 – 253.

[24] Gold E M. Language identification in the limit[J]. Information and control,1967,10(5):447 – 474.

[25] Bex G J, Gelade W, Neven F, et al. Learning deterministic regular expressions for the inference of schemas from XML data[A]. Proceedings of the WWW[C]. New York:ACM Press,2008. 825 – 834.

[26] Garofalakis M, Gionis A, Rastogi R, et al. XTRACT: learning document type descriptors from XML document collections[J]. Data Mining And Knowledge Discovery, 2003,7(1):23 – 56.

[27] Min J K, Ahn J Y, Chung C W. Efficient extraction of schemas for XML documents[J]. Information Processing Letters,2003,85(1):7 – 12.

[28] Vošta O, Mlýnková I, et al. Even an ant can create an XSD[A]. Proceedings of the DASFAA[C]. Berlin Heidelberg:Springer,2008. 35 – 50.

[29] 宁静,刘杰,叶丹. 一种基于内容模型图 XML Schema Definition 的提取方法[J]. 计算机科学,2010,37(6):179 – 185.
Ning Jing, Liu Jie, Ye Dan. Novel approach for extracting

XML schema definition based on content model graph [J]. Computer Science, 2010, 37(6): 179 – 185. (in Chinese)

[30] Nierman A, Jagadish H V. Evaluating structural similarity in XML documents[A]. Proceedings of the WebDB[C]. New York: ACM Press, 2002. 61 – 66.

[31] Jain A K, Dubes R C. Algorithms for Clustering Data [M]. New York: Prentice-Hall Inc., 1988. 55 – 142.

[32] Moh C H, Lim E P, Ng W K. Re-engineering structures from Web documents[A]. Proceedings of the fifth ACM conference on Digital Libraries [C]. New York: ACM Press, 2000. 67 – 76.

[33] Grünwald P. Model selection based on minimum description length [J]. Journal of Mathematical Psychology, 2000, 44(1): 133 – 152.

[34] Sankey J, Wong R K. Structural inference for semistructured data [A]. Proceedings of the CIKM [C]. New York: ACM Press, 2001. 159 – 166.

[35] Dorigo M, Birattari M, Stutzle T. Ant colony optimization [J]. IEEE Computational Intelligence Magazine, 2006, 1(4): 28 – 39.

[36] Zhang Y, Zhou H, Liu J, et al. Efficient schema extraction from large XML documents[A]. Proceedings of the 5th International Conference on Biomedical Engineering and Informatics[C]. Berlin: IEEE Computer Society Press, 2012. 1255 – 1260.

[37] Hegewald J, Naumann F, Weis M. XStruct: efficient schema extraction from multiple and large XML documents [A]. Proceedings of the ICDE Workshops[C]. Berlin: IEEE Computer Society Press, 2006.

[38] Chidlovskii B. Schema extraction from XML: a grammatical inference approach [A]. Proceedings of the KRDB [C]. Aachen Germany: CEUR-WS. org, 2001.

[39] Mlýnková I, Nečaský M. Towards inference of more realistic XSDs[A]. Proceedings of the SAC[C]. New York: ACM Press, 2009. 639 – 646.

[40] Kozák M, Stárka J, Mlýnková I. Schematron schema inference[A]. Proceedings of the IDEAS[C]. New York: ACM Press, 2012. 42 – 50.

[41] Ciucanu R, Staworko S. Learning schemas for unordered XML[A]. Proceedings of the DBPL[C]. New York: ACM Press, 2013. 31 – 40.

[42] Mlýnková I, Nečaskčý M. Heuristic methods for inference of XML schemas: lessons learned and open issues[J]. Informatica, 2013, 24(4): 577 – 602.

[43] Klempa M, et al. jInfer: A framework for XML schema inference[J]. The Computer Journal, to appear. doi: 10.1093/comjnl/bxt148.

[44] Bex G J, Neven F, Bussche J. DTDs versus XML Schema: a practical study [A]. Proceedings of the WebDB [C]. New York: ACM Press, 2004. 79 – 84.

[45] Bex G J, et al. Inference of concise DTDs from XML data [A]. Proceedings of the VLDB [C]. New York: ACM Press, 2006. 115 – 126.

[46] Bex G J, Neven F, Schwentick T, et al. Inference of concise regular expressions and DTDs[J]. ACM Transactions on Database Systems, 2010, 35(2): 1 – 47.

[47] Garcia P, Vidal E. Inference of *k*-testable languages in the strict sense and application to syntactic pattern recognition [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1990, 12(9): 920 – 925.

[48] Freydenberger D, Kötzing T. Fast learning of restricted regular expressions and DTDs [A]. Proceedings of the ICDT[C]. New York: ACM Press, 2013. 45 – 56.

[49] Freydenberger D, Reidenbach D. Inferring descriptive generalisations of formal languages[J]. Journal of Computer and System Sciences, 2012, 79(5): 622 – 639.

[50] Bex G J, Gelade W, Neven F, et al. Learning deterministic regular expressions for the inference of schemas from XML data[J]. ACM Transactions on the Web, 2010, 4(4): 14: 1 – 14: 32.

[51] Bex G J, Neven F, Vansummeren S. Inferring XML schema definitions from XML data[A]. Proceedings of the VLDB[C]. New York: ACM Press, 2007. 998 – 1009.

[52] Barbosa D, Mendelzon A. Finding ID attributes in XML documents[J]. Lecture Notes in Computer Science, 2003, 2824: 180 – 194.

[53] Vitásek M, Mlynková I. Inference of XML integrity constraints[J]. Advances in Databases and Information Systems, 2013, 186: 285 – 296.

[54] Arenas M, Daenen J, Neven F, et al. Discovering XSD keys from XML data[A]. Proceedings of the SIGMOD [C]. New York: ACM Press, 2013. 61 – 72.

[55] MannilaH, Toivonen H. Levelwise search and borders of theories in knowledge discovery [J]. Data mining and knowledge discovery, 1997, 1(3): 241 – 258.

[56] Mannila H, Räihä K J. Algorithms for inferring functional dependencies from relations[J]. Data & Knowledge Engineering, 1994, 12(1): 83 – 99.

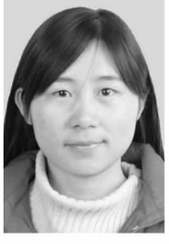
[57] Mannila H, Raiha K J. Practical algorithms for finding prime attributes and testing normal forms [A]. Proceedings of the PODS[C]. New York: ACM Press, 1989. 128 – 133.

[58] Bitton D, Millman J, Torgersen S. A feasibility and performance study of dependency inference[A]. Proceedings of the ICDE[C]. Berlin: IEEE Computer Society Press, 1989. 635 – 641.

[59] Buneman P, Davidson S, Fan W, et al. Keys for XML

[J]. Computer Networks,2002,39(5):473 – 487.
[60] Fajt S,Mlynkova I,Necasky M. On mining XML integrity constraints[A]. Proceedings of the 6th International Conference on Digital Information Management[C]. Berlin: IEEE Computer Society Press,2011. 23 – 29.
[61] Grahne G,Zhu J. Discovering approximate keys in XML data[A]. Proceedings of the CIKM. [C]. New York: ACM Press,2002. 453 – 460.
[62] Liu Y, Ye F, He S. Mining approximate keys based on reasoning from XML data[A]. Proceedings of the PAKDD Workshops[C]. Berlin Heidelberg: Springer, 2013. 499 – 510.
[63] Nečaský M,Mlýnková I. Discovering XML keys and foreign keys in queries[A]. Proceedings of the SAC[C]. New York:ACM Press,2009. 632 – 638.
[64] Trinh T. Using transversals for discovering XML functional dependencies[J]. Lecture Notes in Computer Science, 2008,4932:199 – 218.
[65] Shi H, Amagasa T, Kitagawa H. Fast detection of functional dependencies in XML data[J]. Lecture Notes in Computer Science,2010,6309:113 – 127.
[66] Liu J, et al. Discover dependencies from data-a review [J]. IEEE Transactions on Knowledge and Data Engineering,2012,24(2):251 – 264.
[67] 金峰,陶晓鹏,胡运发. XML 函数约束规则的自动挖掘[J]. 计算机科学,2003,30(10):227 – 229.
Jin Feng,Tao Xiaofeng,Hu Yunfa. Automatica mining of functional constraint rule in XML document[J]. Computer Science,2003,30(10):227 – 229. (in Chinese)

作者简介



郑黎晓　女,1983 年 9 月生于河南南阳. 分别于 2006 年和 2012 年于吉林大学计算机科学与技术学院和中国科学院软件研究所获学士学位和博士学位. 现为华侨大学计算机科学与技术学院讲师. 研究方向:形式语言与自动机,数据库理论,软件测试等.
E-mail:zhenglx@ hqu. edu. cn

王　成　男,1984 年 4 月生. 分别于 2006 年和 2012 年于电子科技大学和西安交通大学获学士学位和博士学位. 现为华侨大学计算机科学与技术学院讲师. 研究方向:机器学习,智能电子商务数据挖掘.
E-mail:wangcheng@ hqu. edu. cn